**import** java.applet.\*;

**import** java.awt.\*;

**import** java.util.\*;

**import** java.text.\*;

**public** **class** MyClock **extends** Applet **implements** Runnable {

**int** width, height;

   Thread t = **null**;

**boolean** threadSuspended;

**int** hours=0, minutes=0, seconds=0;

   String timeString = "";

**public** **void** init() {

      width = getSize().width;

      height = getSize().height;

      setBackground( Color.black );

   }

**public** **void** start() {

**if** ( t == **null** ) {

         t = **new** Thread( **this** );

         t.setPriority( Thread.MIN\_PRIORITY );

         threadSuspended = **false**;

         t.start();

      }

**else** {

**if** ( threadSuspended ) {

            threadSuspended = **false**;

**synchronized**( **this** ) {

               notify();

            }

         }

      }

   }

**public** **void** stop() {

      threadSuspended = **true**;

   }

**public** **void** run() {

**try** {

**while** (**true**) {

            Calendar cal = Calendar.getInstance();

            hours = cal.get( Calendar.HOUR\_OF\_DAY );

**if** ( hours > 12 ) hours -= 12;

            minutes = cal.get( Calendar.MINUTE );

            seconds = cal.get( Calendar.SECOND );

            SimpleDateFormat formatter

               = **new** SimpleDateFormat( "hh:mm:ss", Locale.getDefault() );

            Date date = cal.getTime();

            timeString = formatter.format( date );

            // Now the thread checks to see if it should suspend itself

**if** ( threadSuspended ) {

**synchronized**( **this** ) {

**while** ( threadSuspended ) {

                     wait();

                  }

               }

            }

            repaint();

            t.sleep( 1000 );  // interval specified in milliseconds

         }

      }

**catch** (Exception e) { }

   }

**void** drawHand( **double** angle, **int** radius, Graphics g ) {

      angle -= 0.5 \* Math.PI;

**int** x = (**int**)( radius\*Math.cos(angle) );

**int** y = (**int**)( radius\*Math.sin(angle) );

      g.drawLine( width/2, height/2, width/2 + x, height/2 + y );

   }

**void** drawWedge( **double** angle, **int** radius, Graphics g ) {

      angle -= 0.5 \* Math.PI;

**int** x = (**int**)( radius\*Math.cos(angle) );

**int** y = (**int**)( radius\*Math.sin(angle) );

      angle += 2\*Math.PI/3;

**int** x2 = (**int**)( 5\*Math.cos(angle) );

**int** y2 = (**int**)( 5\*Math.sin(angle) );

      angle += 2\*Math.PI/3;

**int** x3 = (**int**)( 5\*Math.cos(angle) );

**int** y3 = (**int**)( 5\*Math.sin(angle) );

      g.drawLine( width/2+x2, height/2+y2, width/2 + x, height/2 + y );

      g.drawLine( width/2+x3, height/2+y3, width/2 + x, height/2 + y );

      g.drawLine( width/2+x2, height/2+y2, width/2 + x3, height/2 + y3 );

   }

**public** **void** paint( Graphics g ) {

      g.setColor( Color.gray );

      drawWedge( 2\*Math.PI \* hours / 12, width/5, g );

      drawWedge( 2\*Math.PI \* minutes / 60, width/3, g );

      drawHand( 2\*Math.PI \* seconds / 60, width/2, g );

      g.setColor( Color.white );

      g.drawString( timeString, 10, height-10 );

   }

}